

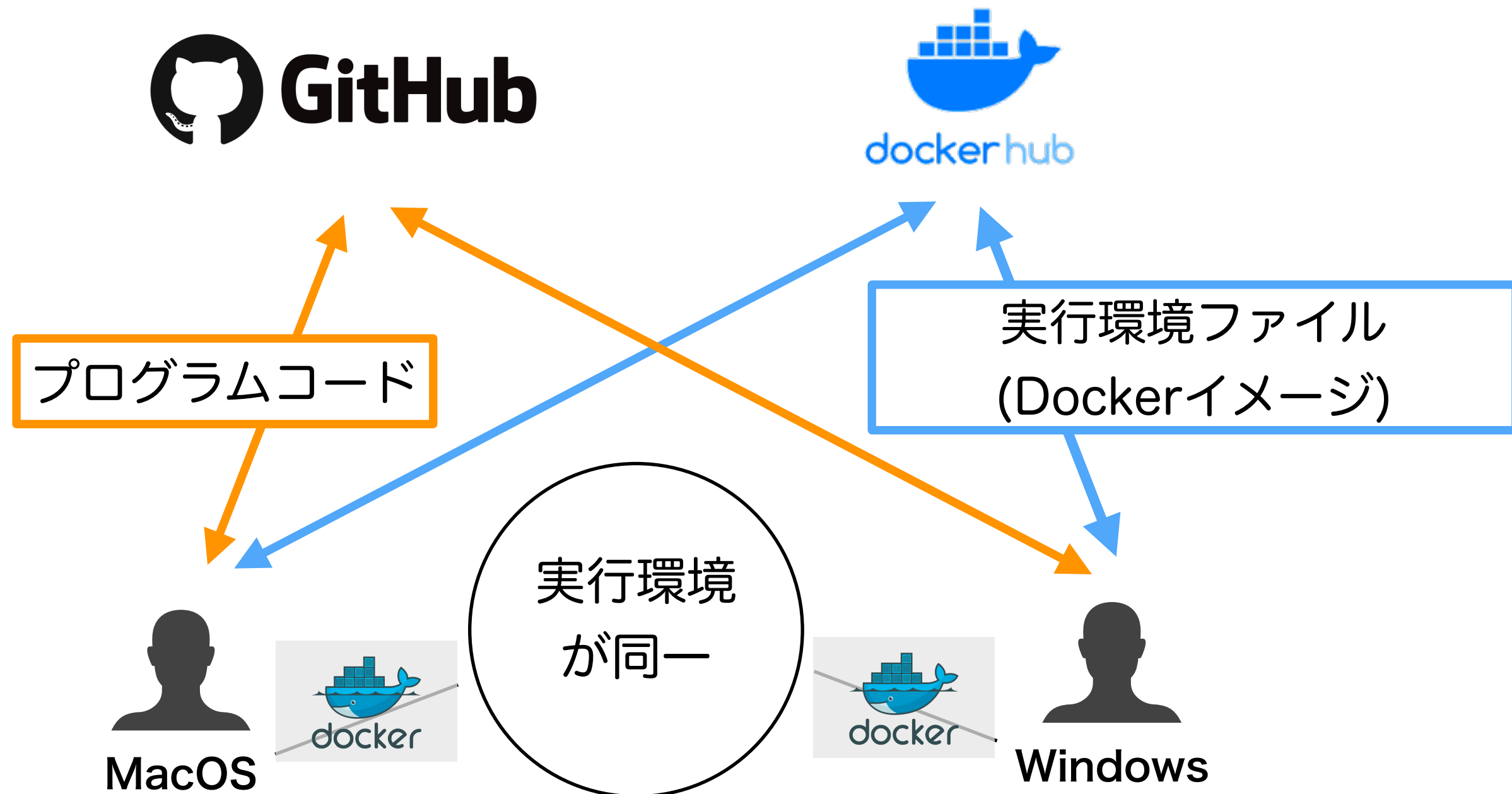
Dockerイメージサイズの削減に むけたIssue要約ツールの提案

オープンソースソフトウェア工学研究室
M2 60210045 作島大智

背景 | 対象技術

Docker

軽量なコンテナ型の仮想環境を
作成・配布・実行するためのプラットフォーム



背景

- ソフトウェア開発において開発者はDockerを盛んに利用している
 - 正確に素早く開発者間で開発環境を共有できるため
- プロジェクトは実行環境ファイルであるDockerイメージのサイズを小さくしたいと考えている
 - ex(Azure*等のクラウドサービスを用いる場合,
イメージサイズとビルド時間が大きくなるほど料金が高くなる
- 近年ではイメージサイズがどんどん小さくなっている [1]



* <https://azure.microsoft.com/ja-jp/>

[1] Lin, Changyuan & Nadi, Sarah & Khazaei, Hamzeh. (2020).
A Large-scale Data Set and an Empirical Study of Docker Images Hosted on Docker Hub.

背景

- イメージサイズを小さくするためにはイメージ内に含まれる
不必要なパッケージを削減する必要がある
- 公式Dockerイメージは大きく2種類存在する

通常
イメージ

一般的なパッケージ全てが
含まれるイメージ

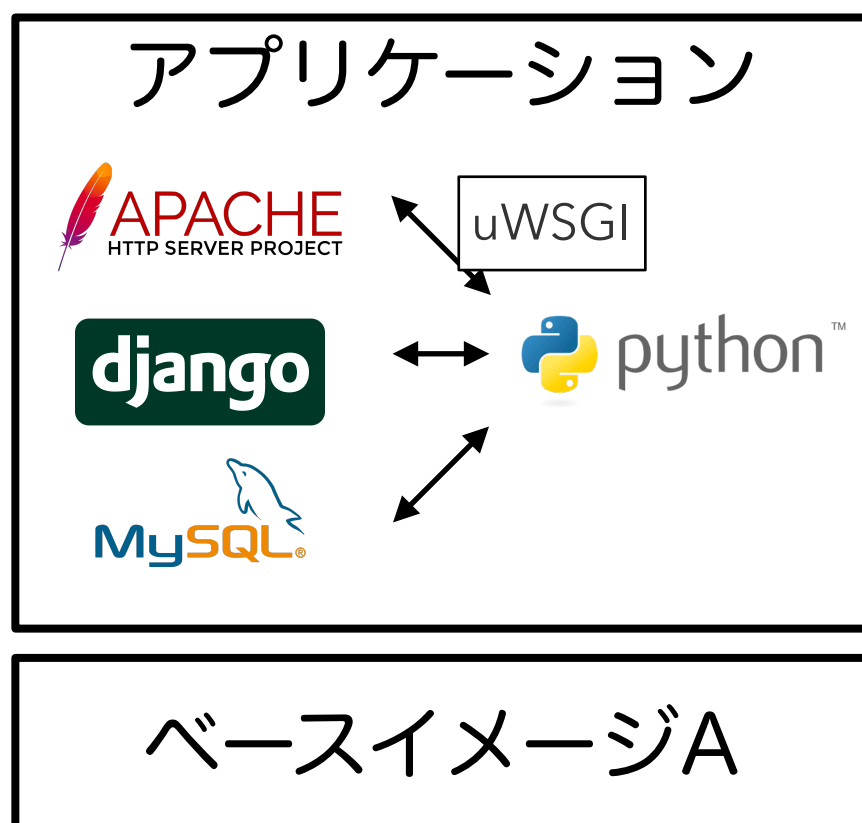
スリム
イメージ

必要最低限のパッケージが
含まれるイメージ

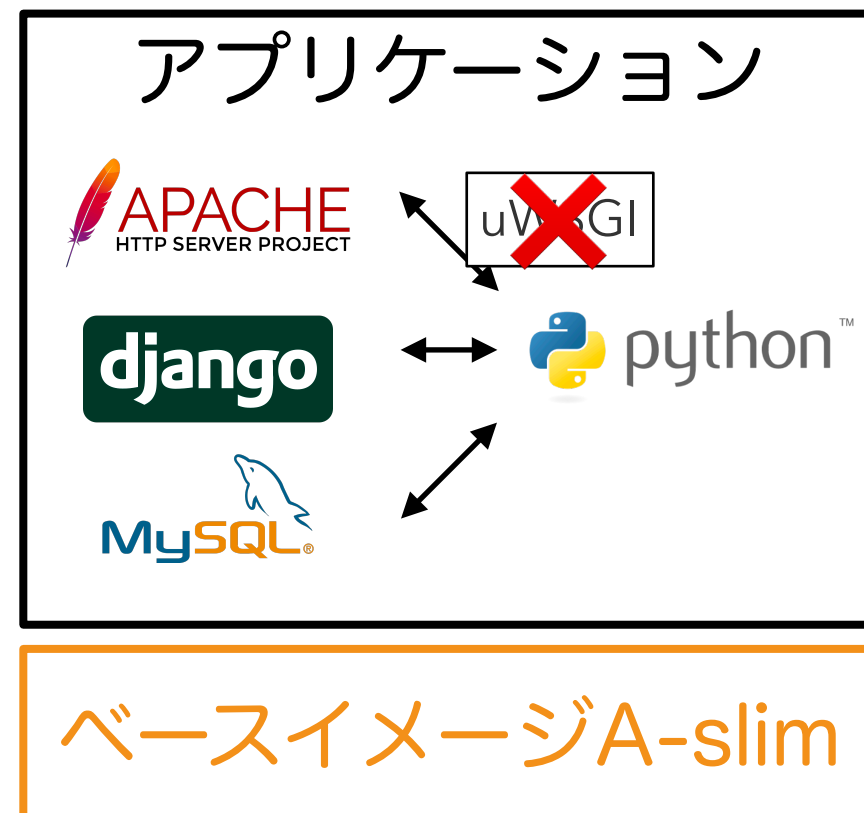
イメージサイズに関心のある利用者は主にスリムイメージを使用する

イメージサイズを削減する際の課題 | 具体例

○ webアプリ開発の際, uWSGI が必要になった場合



ベースイメージAであれば
問題なくインストールできる



ベースイメージA-slimにはコンパイラがない
→ コンパイラをインストールする必要あり

利用者が求める機能を持つ小さいDockerイメージを作成するには
Dockerイメージをカスタマイズする必要がある

イメージサイズを削減する際の課題

- Dockerイメージをカスタマイズする
= 必要なパッケージと不要なパッケージを判別する
- スリムイメージのパッケージに関する体系的なドキュメントはない
→ エラー等が生じた際に対処が困難
- エラーを解決するためには関連するソースを調査し、
自身で情報を収集する必要がある

課題

パッケージに関して調査する時間的コストが
非常に大きい

先行研究 | 不具合票の要約生成

- バグ報告内の議論は多く，効率的に必要な情報を知りたい
 - 議論の場であり，後から見返すように設計されていない
- 今まで多くのバグ報告の要約の研究が行われている
 - 教師あり or 教師なしの機械学習で実装
- 従来の要約手法で生成される要約文は冗長で信頼性に欠ける
 - 頻出単語に依存するため
 - コメントの信憑性を加味していない

→ **新たな教師なしバグ報告要約法の提案[2]**

[2] Haoran Liu, Yue Yu, Shanshan Li, Yong Guo, Deze Wang, and Xiaoguang Mao. 2020.

BugSum: Deep Context Understanding for Bug Report Summarization.

In Proceedings of the 28th International Conference on Program Comprehension (ICPC '20).

Association for Computing Machinery, New York, NY, USA, 94-105. DOI:<https://doi.org/10.1145/3387904.3389272>

先行研究 | 課題点

- パッケージ不足issueに対して正確に要約文を生成できない可能性がある
 - パッケージ不足issueは複雑であるため議論が発散する
 - 簡潔な要約文が生成できない
- より抽象的な特徴量を用いて、要約の冗長性を防ぐ
 - 解決策部分に特化して文を抜き出すことで必要な情報を簡潔にまとめる

着眼点

文の句構造の特徴を用いて要約文を選択する

本研究の目的

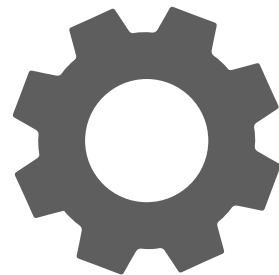
目的

Dockerのパッケージ不足に関する情報を
効率よく多く集めること

→ パッケージ不足issue要約ツールの提案
(元の文章から重要な文章を抜き出す抽出型)



要約ツール



パッケージ不足issue

問題部分, 解決方法の文章を
抽出する

Issue (課題管理票)

- GitHub(共同開発環境)に実装されている機能
- 開発製品の不具合や機能改善要求などの様々な議論内容を管理

CVE-2021-23840 still in recent Debian Buster based Python images #580 タイトル

Closed ahaerpfer opened this issue on 21 Feb · 2 comments

質問者 ahaerpfer commented on 21 Feb

Is it possible that the most recent builds for the Debian Buster based images were built from an outdated Debian package cache? Dockerfiles for the 3.8 and 3.9 versions were updated on Feb. 19 ([c285e28](#) , [91cbd74](#)), but the corresponding images on Dockerhub still contain the vulnerable `openssl 1.1.1d-0+deb10u4`.

On the other hand the [Debian Security Tracker](#) lists this vulnerability as fixed with the updated `openssl 1.1.1d-0+deb10u5` being available since Feb. 17 ([DSA-4855-1](#)), i.e. two days earlier!

When I build a `python:3.9-slim` image from the Dockerfile in this repo myself (today, i.e. Feb 20), I get the updated package.

See also [#578](#).

本文

コメント ahaerpfer changed the title ~~CVE-2021-23840 still in recent Debian Buster images~~ CVE-2021-23840 still in recent Debian Buster based Python images on 21 Feb

回答者 wglambert commented on 23 Feb

Looks like the recent security update just missed it [docker-library/official-images#9658](#)

But there's another PR up [docker-library/official-images#9670](#)

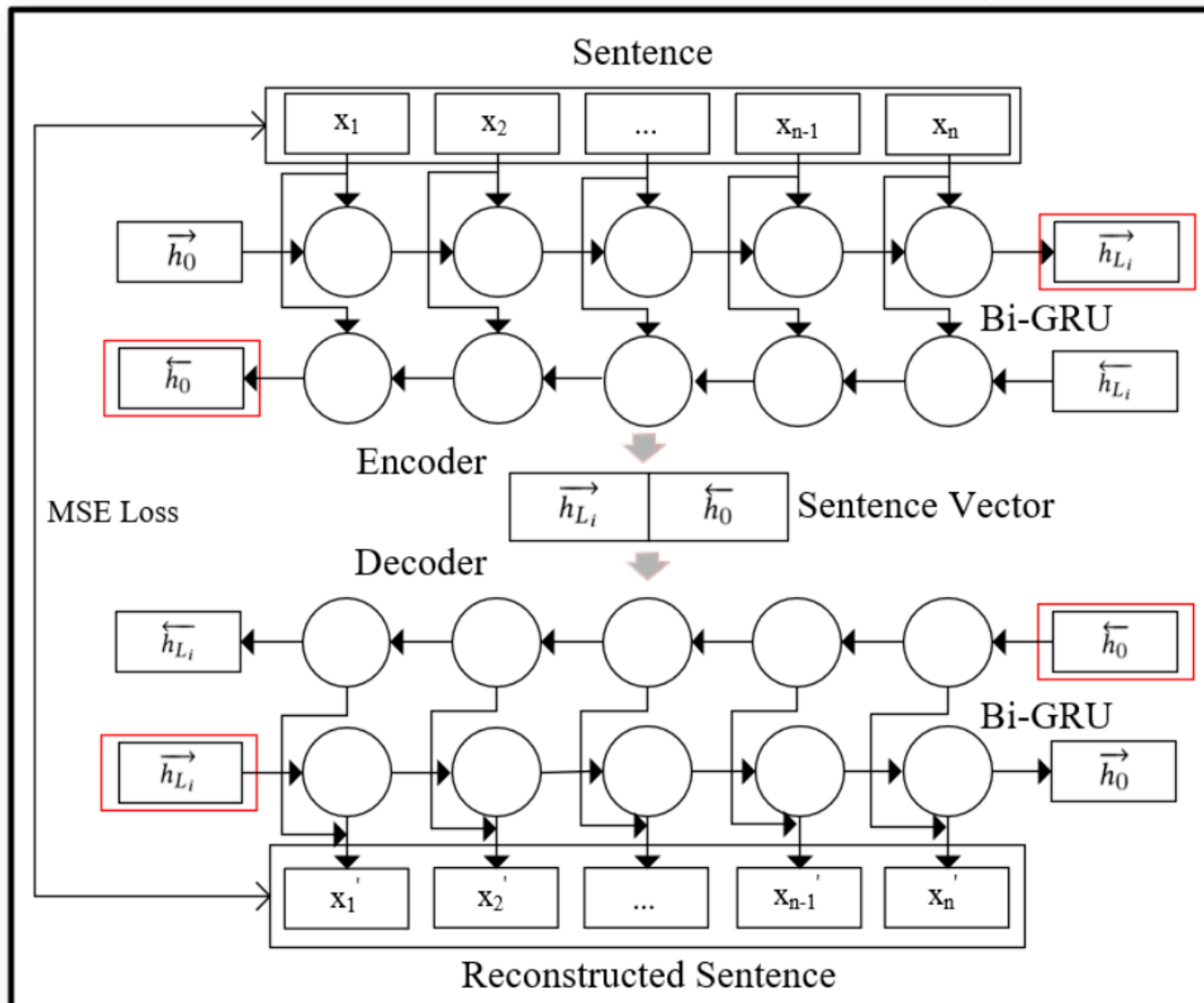
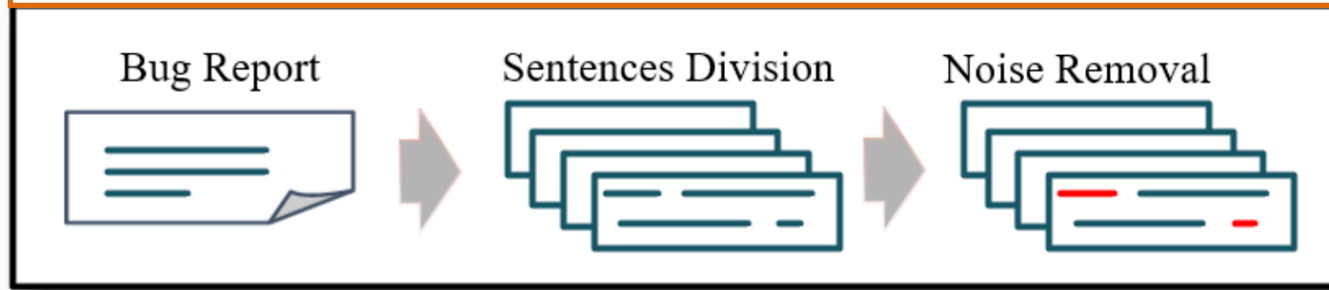
コメント yosifkit commented on 23 Feb

Unfortunately, [docker-library/official-images#9670](#) is not going to have an effect, since it only changes the pip installation. That happens after the installation of python and the builds make heavy use of Docker build cache, so the earlier layers will be unchanged when the images are built.

コメント

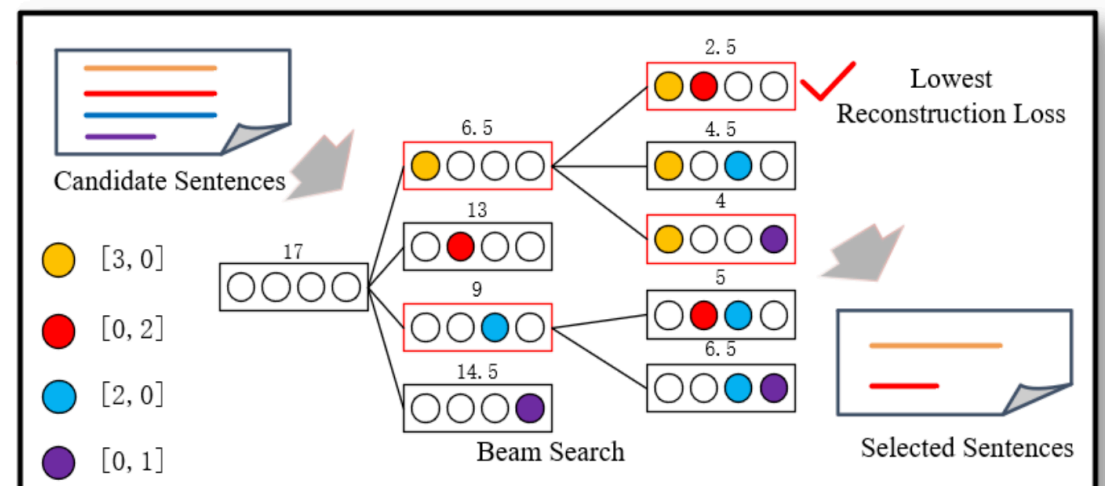
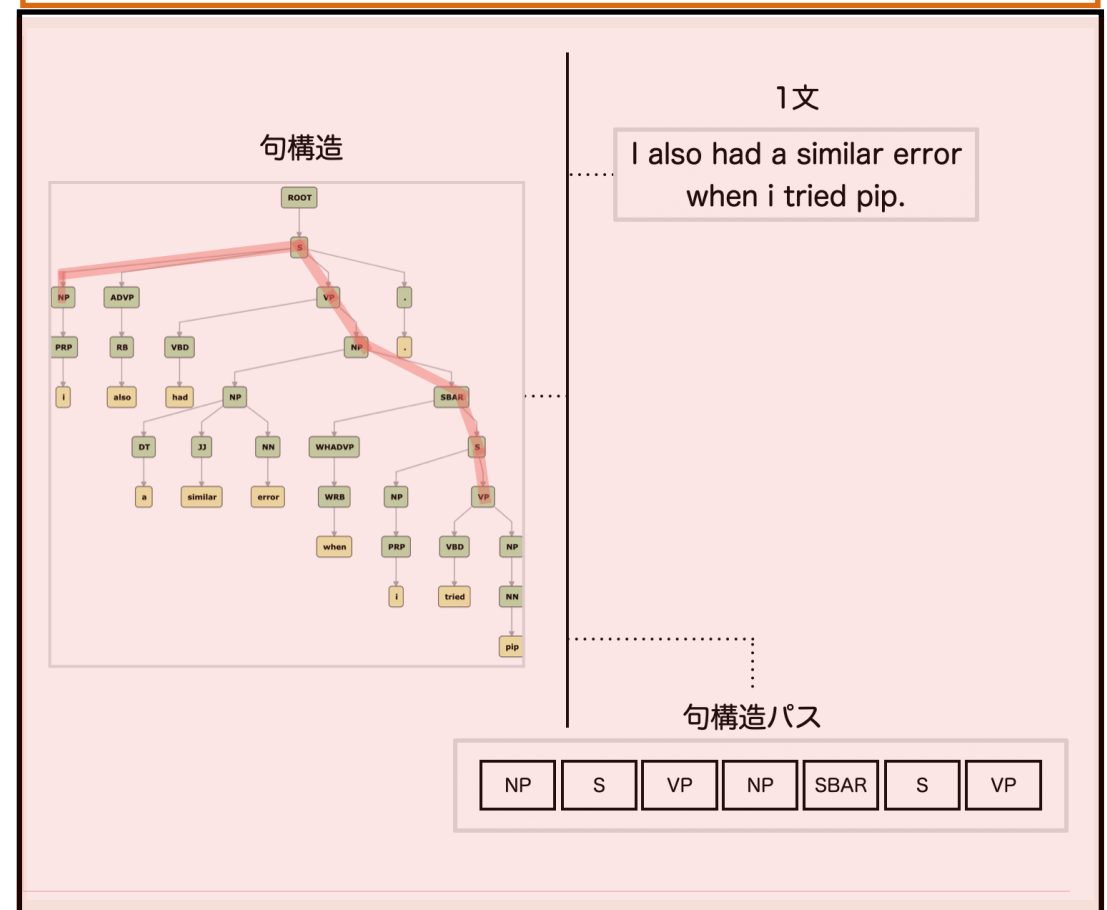
提案ツールの概要

issueの前処理



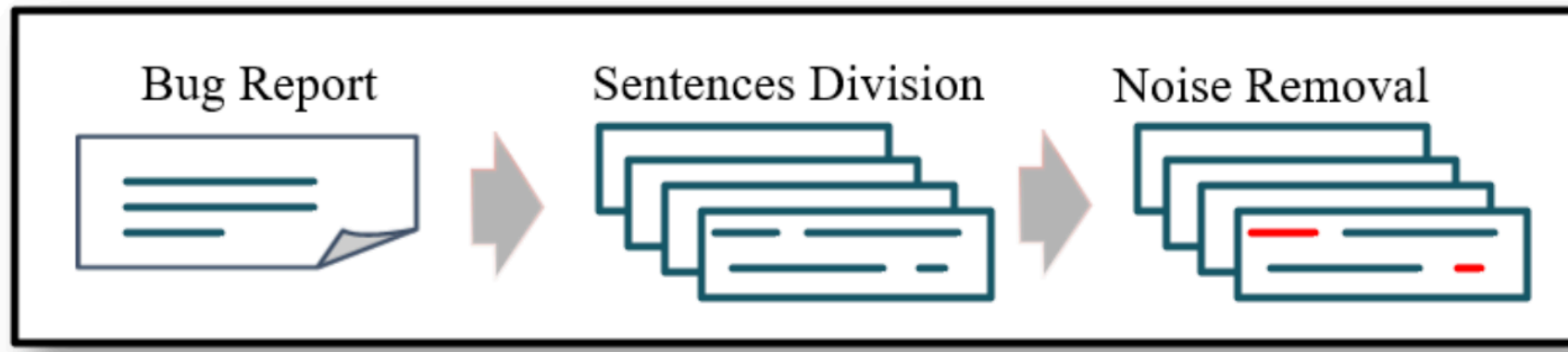
テキスト情報量の抽出

文の句構造を用いてスコアを変化させる



最も数値の高い文 = 解決策

提案ツール | テキストデータの前処理



- 文字列の一部として句読点を使用している場合以外削除
 - ソフトウェア固有の正規表現[2] ($([\backslash w-]+(\backslash.([\backslash w-]+)*)$) を使用して文章をトークン化
 - 単語を正しく識別しながら関数名や変数名の大部分を保持する
- poster stemmer[3]を用いてステム処理
- ストップワード[4]処理

[2] Rafael Lotufo, Zeeshan Malik, and Krzysztof Czarnecki. 2015.

Modelling the 'hurried' bug report reading process to summarize bug reports. Empirical Software Engineering 20, 2 (2015), 516-548.

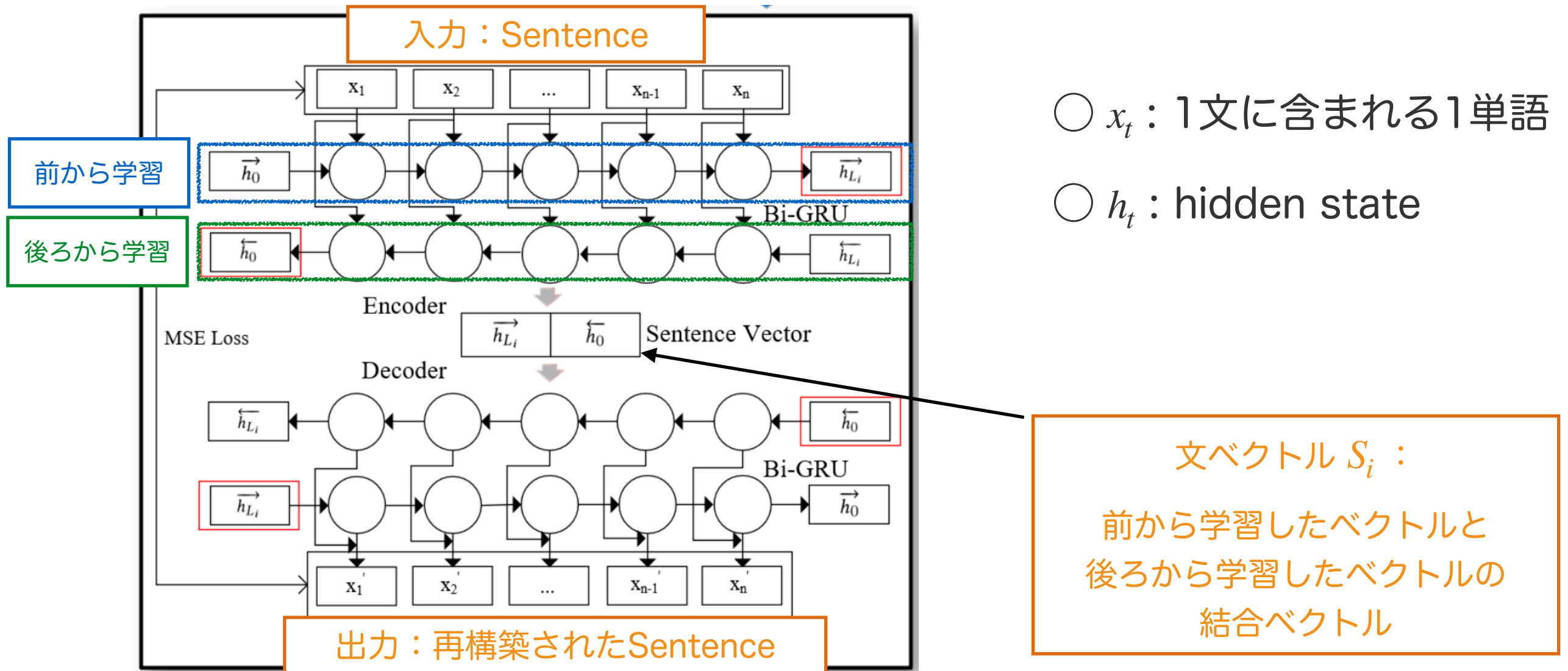
[3] Martin F Porter. 1980. An algorithm for suffix stripping. Program 14, 3 (1980), 130-137.

[4] Damian Doyle. [n.d.]. Default English stopwords list. <https://www.ranks.nl/stopwords>. 2017.

提案ツール | テキスト特徴量の抽出

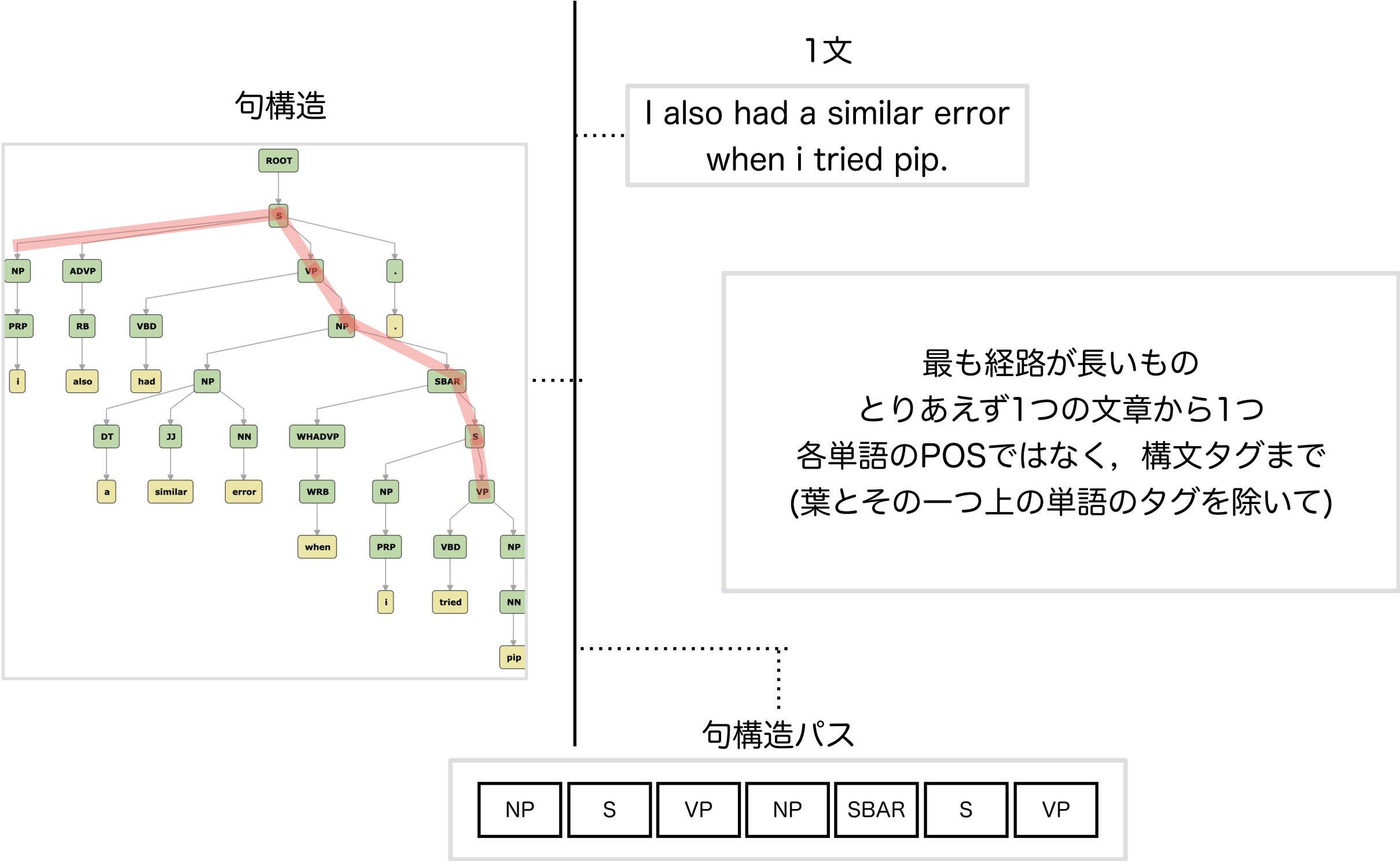
○ オートエンコーダ

- リカレントユニットには双方向GRU (Bi-GRU) [51]を採用 ← LSTMの拡張版



MSEが最小になる S_i = 文章の特徴を正確に表現している
(学習率0.01に設定)

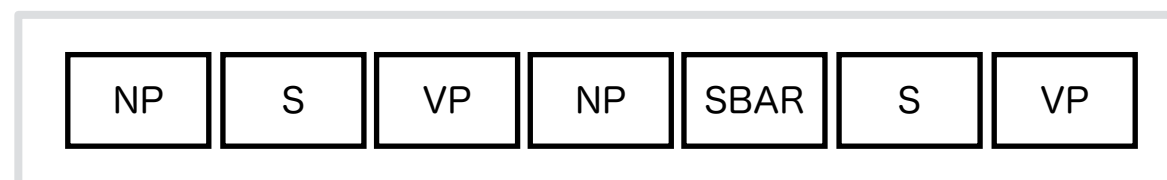
提案ツール | 句構造パスの抽出



※ 上記のPOSは構文タグを示す
ex) NP:名詞句, VP:動詞句, SBAR:従属節句

提案ツール | 句構造パスの抽出

1文の句構造パス



× 文の個数分



クラスタリング

解決策部分

問題部分

要因部分

その他部分

それぞれを特徴づける句構造パスを抽出



秀出した句構造パスの特徴を加味して要約選択を行う

現在、句構造パスの検証を行なっている
(句構造パスの選択方法, どのようにベクトル化するか等)

提案手法 | スコアリング + 要約生成

○ 文 s_i のスコアリング

- オートエンコーダで抽出した情報量 + 句構造パス

○ 要約生成

- バグ報告の文全ての特徴量と選択した文の特徴量の平均二乗誤差が最小になる組み合わせを算出する

○ 生成した要約文の評価

- パッケージ不足issueに対応する要約を作成し、それを正解集合とする
- 先行研究との比較にはベンチマークデータセットを用いる？

未定

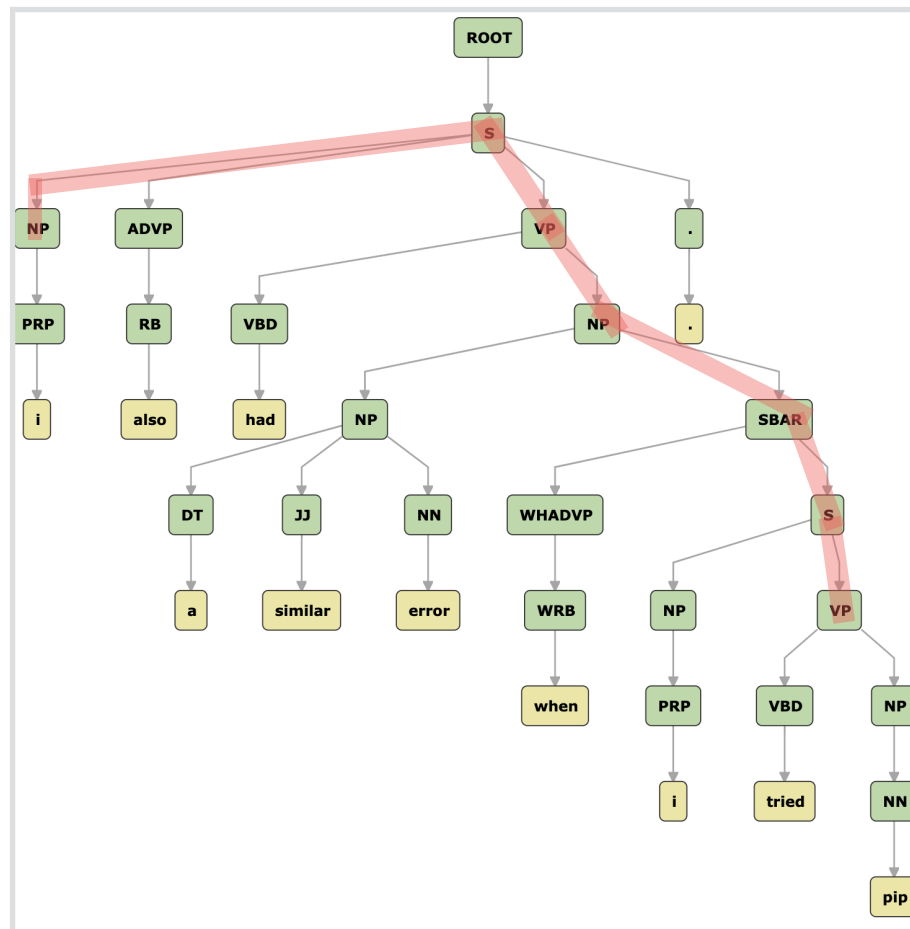
現在していること

○ 句構造パスの検証

- 最も長いパスだけでいいのか
 - 順番を考慮する必要があるのか
 - どのようにベクトル化するのか
- 構造的にどのように異なるのか

付録

句構造



1文

I also had a similar error
when i tried pip.

句構造パス

NP	S	VP	NP	SBAR	S	VP
----	---	----	----	------	---	----